ICC - *STRINGS E FUNÇÕES*

PROF. FERNANDO W CRUZ

Roteiro da aula

- Atividade prática - geração de relatório usando funções e manipulação de *strings*

Problema:

Todos os semestres ingressam 240 alunos no curso de Engenharia da FGA e cada aluno precisa cursar algumas disciplinas obrigatórias, dentre elas a disciplina de ICC. Organize um programa em linguagem C para imprimir a relação de alunos e notas obtidas no semestre para essa disciplina (considere que todos os alunos cursam essa disciplina). A impressão deve seguir o seguinte formato:

UnB - Universidade de Brasília / FGA-Gama
ICC - Introdução à Ciência da Computação
1o. semestre de 2010

>> RELAÇÃO DE ALUNOS E NOTAS OBTIDAS NO SEMESTRE <<

ALUNO	MÉDIA
ANTÔNIA MARIA NOGUEIRA	45
BRUNO CASTANHEIRAS	62
CARLOS ALBERTO DA FONSECA	100
JULIA BELTRÃO	53
.	.
.	.
.	.
ZÉLIA CARDOSO SENNA AIRES	70
Média da turma:	83
Número de alunos que ficaram acima da média:	150

Pede-se:

Para atender ao que foi solicitado, o programa deve ainda ser composto pelo seguinte grupo de funções:

a)Uma função para ler os nomes à partir do teclado. Obs.: Os nomes podem ter tamanhos variados, podem ser digitados com letras maiúsculas e minúsculas e não serem necessariamente digitados em ordem alfabética. Assumir também que nenhum nome possui cedilhas, acentos e outros caracteres diferentes dos que estão apresentados na Tabela ASCII na página seguinte.

b)Uma função para ler as notas dos alunos. Essas notas podem ir de zero a 100 (números inteiros).

c)Uma função para calcular a média da turma.

d)Uma função para calcular o número de alunos que possuem nota maior do que a média da turma (calculada na função anterior).

e)Uma função para converter os nomes dos alunos em letras maiúsculas.

f)Uma função para ordenar os nomes dos alunos apenas pela primeira letra do nome. Portanto, a solicitação é que apenas o primeiro caracter da matriz de nomes seja considerada na ordenação

Requisitos a serem cumpridos:

- ✓Deve-se assumir que os alunos possuem nomes menores do que 50 caracteres.
- ✓Para o item *(a)*, usar o comando *getchar()* para leitura dos nomes.
- ✓Em relação às estruturas de repetição do programa: *(i)* Para o item *(a)*, usar o do-while(); *(ii)* Para o item *(d)*, usar o comando *for* ; *(iii)* Para os demais itens, usar o comando *while*
- ✓As funções das letras *(d)* e *(e)* devem estar depois do programa principal (função *main()*); as demais ficam antes do programa principal.
- ✓Usar identação adequada para os comandos.
- ✓Gerar documentação adequada, ou seja, colocar a identificação do aluno (matrícula e nome) e colocar comentários que descrevam o que cada função faz.
- ✓Usar nomes significativos para as variáveis usadas no programa

Tabela ASCII

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

Solução: programa principal (sugestão):

```
#include <stdio.h>
#define TAM_TURMA     3
#define TAM_NOME      50

main () {
   char icc_nomes[TAM_TURMA][TAM_NOME];
   int  icc_notas[TAM_TURMA];
   int i;
   float media;
   int tot_alunos_acima_media;

   i=0;
   while (i<TAM_TURMA) {
      leia_nome(icc_nomes[i], i);
      icc_notas[i] = leia_nota(i);
      i++;
   } /* fim-while */
//media = calcula_media(icc_notas);
//tot_alunos_acima_media = calcula_total_acima_media(icc_notas, media);
//converte_maiuscula(icc_nomes);
//ordena_nomes(icc_nomes, icc_notas);
   imprime_relatorio(icc_nomes, icc_notas, media, tot_alunos_acima_media);
} /* fim-main */
```

Declaração das estruturas de dados necessárias

Chamadas para as funções de leitura de nomes e notas

Solução: funções para leitura de nomes e notas:

```
void leia_nome (char s[TAM_NOME], int aluno) {
  int i, c;
  i = 0;
  printf("Digite o nome do aluno %d : ", aluno);

  c = getchar();
  if (c == '\n') {
     c = getchar();
  }
  do  {
     s[i] = c;
     i++;
     c = getchar();
  } while (c != '\n'); /* fim-do-while */
  s[i] = '\0';
} /* fim-leia_nome */

int leia_nota (int aluno) {
  int nota;
  printf("Digite a nota do aluno %d : ", aluno);
  scanf("%d", &nota);
  return (nota);
} /* fim-leia_nota */
```

Elimina “enters” de digitações anteriores

Preenchimento do vetor s (linha da matriz icc_nomes)

Solução: funções para leitura de nomes e notas:

```
float calcula_media(int s[]) {
  float med;
  int soma = 0;
  int i=0;

  while (i<TAM_TURMA) {
     soma = soma + s[i];
     i++;
  } /* fim-while */
  med = (float) soma / TAM_TURMA;
  return (med);
} /* fim-calcula_media */
```

Laço para acumular os valores do vetor icc_notas (endereçado pelo vetor s)

Cast para garantir a divisão de inteiros, gerando um número real

Solução: função para cálculo da quantidade de alunos acima da média:

```
int calcula_total_acima_media(int s[], float media){
  int i;
  int tot_alunos=0;

  for (i=0;i<TAM_TURMA;i++) {
     if ( (float) s[i] > media) {
        tot_alunos++;
     } /* fim-if */
  } /* fim-for */
  return (tot_alunos);
} /* fim-calcula_total_acima_media */
```

Solução: função para converter letras minúsculas em letras maiúsculas:

```
void converte_maiuscula(char nome_aluno[] [TAM_NOME]) {
  int i, j;

  i=0;
  while (i<TAM_TURMA) {
     j = 0;
     while (nome_aluno[i][j] != '\0') {
        if ((nome_aluno[i][j] >= 97) && (nome_aluno[i][j] <= 122)) {
           nome_aluno[i][j] = nome_aluno[i][j] - 32;
        } /* fim-if */
        j++;
     } /* fim-while */
     i++;
  } /* fim-while */
} /* fim-converte_maiuscula */
```

Veja uma forma possível para receber uma matriz como parâmetro da função.

nome_aluno aqui é uma referência à matriz icc_nomes do programa principal.

Solução: função ordenar a primeira coluna da matriz de nomes:

```
void ordena_nomes(char nome_aluno[][TAM_NOME], int nota_aluno[]) {
  char   aux[TAM_NOME];
  int    auxnota;
  int    lsup, i;

  lsup = TAM_TURMA - 1;
  while (lsup > 0) {
      i =0;
      while (i<lsup) {
         if (nome_aluno[i][0] > nome_aluno[i+1][0]) {
            copia(aux, nome_aluno[i]);
            copia(nome_aluno[i], nome_aluno[i+1]);
            copia(nome_aluno[i+1], aux);

            auxnota         = nota_aluno[i];
            nota_aluno[i]   = nota_aluno[i+1];
            nota_aluno[i+1]  = auxnota;
         } /* fim-if */
         i++;
      } /* fim-while */
      lsup--;
  } /* fim-while */
} /* fim-ordena_nomes */
```

Solução: função de apoio para viabilizar a cópia de nomes (exigido na função de ordenação):

```
void copia(char to[], char from[]) {
  int i;

  i=0;
  while (from[i] != '\0') {
     to[i] = from[i];
     i++;
  } /* fim-while */
  to[i] = '\0';
} /* fim-copia */
```

Solução: função para impressão do relatório

```
void imprime_relatorio(char nome_aluno[TAM_TURMA][TAM_NOME], int nota_aluno[], float media, int tot_alunos) {
   int i, j, tabulacao;

   system("clear");
   printf("UnB - Universidade de Brasília / FGA-GAMA\n");
   printf("ICC - Introdução à Ciência da Computação\n");
   printf("1o. Semestre de 2010\n");
   printf("     >> RELAÇÃO DE ALUNOS E NOTAS OBTIDAS NO SEMESTRE <<\n");
   printf("-------------------------------------------------------------\n");
   printf("ALUNO                                NOTA\n");
   printf("-------------------------------------------------------------\n");
   i = 0;
   while (i < TAM_TURMA) {
        j=0;
        while(nome_aluno[i][j] != '\0') {
            printf("%c", nome_aluno[i][j]);
            j++;
        } /* fim-while */
        tabulacao = TAM_NOME+6 - j;
        j = 0;
        while(j<tabulacao) {
            printf("%c", ' ');
            j++;
        } /* fim-while */
        printf("%4d\n", nota_aluno[i]);
        i++;
   } /* fim-while */
   printf("-------------------------------------------------------------\n");
   printf("Média da turma  %44.1f\n", media);
   printf("Número de alunos que ficaram acima da média%17d\n", tot_alunos);
   printf("-------------------------------------------------------------\n");
} /* fim-imprime_relatorio */
```

Imprime o nome e descobre o tamanho desse nome. Tabulação é igual a TAM_NOME+6 unidades menos o tamanho do nome impresso.
O laço mais abaixo preenche com espaços em branco de acordo com a variável tabulacao.
Em seguida o nome é impresso na coluna correta.